

Table based KNN Variants for Segmenting Text based on Contents

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the table based KNN variants as the approaches to the text segmentation. The initial KNN algorithm was previously modified into the table-based version and applied to the task with the view of it into a binary classification. In the case of the numerical vector-based versions, we mention the three KNN variants: one where the selected nearest neighbors are discriminated by their similarities, one where the attributes are discriminated by their correlations with the categories, and one where the training examples are discriminated by their credits. The three KNN variants are modified into the table-based versions, as well as the initial KNN version. This research is intended to improve further the text segmentation performance by modifying them so.

I. INTRODUCTION

The text segmentation is defined as the process of partitioning a single text into more than one subtext based on the topic transition. The assumption in this research is that the text segmentation is applied to the entire text by the boundary between paragraphs. The text segmentation is mapped into the binary classification where each adjacent paragraph pair is classified into boundary or continuance, and a machine learning algorithm is applied to the binary classification. A single text is partitioned on the position between paragraphs in the pair which is classified into boundary. The subtexts which are results from segmenting a text are regarded as independent texts; they may be classified into their different categories.

The motivation of this research is the successful case of applying the table-based version to the text segmentation. It was mapped into the task where an adjacent paragraph pair is classified into boundary or continuance, adjacent paragraph pairs were encoded into tables, and the similarity metric between tables was defined. The KNN algorithm was modified into the table-based version, based on the similarity metric as the approach to the text segmentation. The modified KNN algorithm was empirically validated as its better approach than the traditional version in the text segmentation. In this research, we modify the KNN variants into the table-based versions and apply them to the text segmentation.

In this research, we propose that the KNN variants should be modified into the table-based versions and applied to the text segmentation. The similarity metric between tables

which was used for modifying the KNN algorithm in the previous work, was adopted for modifying the KNN variants as the approaches to the text segmentation. The three table-based KNN variants which are proposed in this research are one where the nearest neighbors are discriminated by their distance, one where the table entries are discriminated by their correlations with the target outputs, and one where the training examples are discriminated by their qualities. The KNN variants are modified into the table-based versions for using them for the text segmentation. The text categorization is the process of classifying a text into one among the predefined topics, whereas the text segmentation is the process of classifying two adjacent paragraphs into one of the two categories.

Let us mention some benefits from this research. The main problems in encoding texts into numerical vectors are prevented by encoding texts into tables. The improvement of classification performance is caused by replacing the KNN algorithm by the KNN variants. The additional improvement of classification performance is caused by modifying the KNN variants into the table-based version. This research is intended to provide the more recommendable approaches to the text segmentation.

Let us mention the organization of this paper. In Section II, we explore the previous works which are relevant to this study. In Section III, we describe in detail what is proposed in this research. In Section IV, we mention the significances of this research and the remaining tasks. This paper is composed of the four sections.

II. PREVIOUS WORKS

This section is concerned with the previous works which are relevant to this research. It is intended to expand the style of modifying the KNN algorithm to its three variants as approaches to text segmentation. The directions of exploring previous works are derivation of KNN variants, modification of the standard KNN algorithm into its table-based version applied to the text segmentation, and the previous case of encoding a text into a table. The distinguishable points of this research are derivation of three KNN variants from the standard version, modification of them into their table-based versions, and application of them to the text segmentation

This article is intended to explore previous works for providing the background for this research.

Let us explore the previous works on KNN variants. In 2001, the KNN variant where nearest neighbors are discriminated by their distances from or similarities as a novice item was applied to the text classification by Han et al. [1]. In 2008, MKNN (Modified K Nearest Neighbor) the KNN variants where training examples are discriminated by their validness, was proposed by Parvin et al. [3]. In 2018, the KNN variant where the attributes are discriminated by their correlations were proposed by Nababan and Sitompul [8]. However, this research uses different specific metrics for discriminating attributes, nearest neighbors, and training examples.

Let us explore the previous works on applying the modified version of KNN algorithm to the text segmentation. In 2017, it was initially proposed that the table-based version of KNN algorithm should be applied to the text segmentation by Jo [6]. In 2018, the modified KNN algorithm was validated in the task by Jo [7]. The journal article which describes in detail the modified KNN algorithm and its application to the text segmentation is finally prepared for its publication by Jo [9]. This research is based on the three previous works.

Let us explore the previous works on the table matching algorithm as the early case of encoding a text into a table. In 2008, the table based matching was proposed as an approach to the text classification as the initial case of encoding a text into a table, instead of a numerical vector [2]. In 2011, the table based matching algorithm was registered as a patent by Jo [4]. In 2015, it was upgraded into its more reliable and stable version by Jo [5]. The similarity metric between two tables is provided by the above previous works for modifying the standard KNN algorithm and its variants.

Let us mention the differences of this research from the previous works which were explored above. The KNN variants in the previous works are numerical vector-based versions, and in this research, they will be modified into table-based versions in this research. As the expansion of [9], we modify and adopt the KNN variants for implementing the text segmentation system. The modified versions of KNN variants become more advanced ones, compared with the table based matching algorithm. The modified KNN variants will be described in detail in Section III.

III. PROPOSED WORKS

This section is concerned with what is proposed in this research. In Section III-A, we describe the process of encoding a text into a table and the proposed similarity metric. In Section III-B, we describe the standard version of KNN algorithm where the proposed similarity metric is adopted. In Section III-C, we derive the three variants from the standard version. In Section III-D, we present the system architecture of the text segmentation system.

A. Encoding Process

This section is concerned with the table as the text representation. It is defined as a set of entries, each of which consists of an element and its weight. The table which represents a text consists of entries, each of which is a pair of a word and its weight in the text. The similarity between tables is computed based on shared words as the semantic similarity between texts. This section is intended to describe the process of encoding a text into a table and the process of computing the similarity between tables.

The process of encoding a text into a table is illustrated in Figure 1. It is indexed into a list of words. The weights of words in the text are computed in the text-word weighting; the word weight in the text is a relationship between a word and a text. The entries with their lower weights are removed for downsizing the table. Refer to [2] for studying the entire process in more detail.

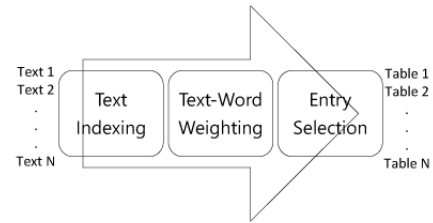


Figure 1. Process of Encoding Texts into Tables

Let us mention the function of a table which maps it into a set of words as shown in Figure 2. The table which represents a text is expressed as entry set as shown in equation (1),

$$T = \{(t_1, w_1), (t_2, w_2), \dots, (t_{|T|}, w_{|T|})\} \quad (1)$$

where t_i is a word, and w_i is the weight of the word, t_i , in the text. The function for generating a list of words is expressed as equation (2),

$$F(T) = \{t_1, t_2, \dots, t_{|T|}\} \quad (2)$$

The function is the operation which takes only words without their weights as a set. The operation is applied to computing the similarity between tables which represent texts.



Figure 2. Conversion of Table into Word Set

Let us mention the process of computing the similarity between two tables as the similarity between two texts. The tables which represent texts are notated by the two

sets: $D_1 = \{(t_{11}, w_{11}), (t_{12}, w_{12}), \dots, (t_{1|D_1|}, w_{1|D_1|})\}$ and $D_2 = \{(t_{21}, w_{21}), (t_{22}, w_{22}), \dots, (t_{2|D_2|}, w_{2|D_2|})\}$. The intersection between the two sets which are generated by applying the function, $F(\cdot)$, by equation (3),

$$F(D_1) \cap F(D_2) = \{t_{s1}, t_{s2}, \dots, t_{s|F(D_1) \cap F(D_2)|}\} \quad (3)$$

and the table with entries each of which consists of a shared word, its weight from the table, D_1 , its weight from the table, D_2 , is constructed based on equation (3), as expressed in equation (4),

$$SD_{12} = \{(t_{s1}, w_{s1}^1, w_{s1}^2), (t_{s2}, w_{s2}^1, w_{s2}^2), \dots, (t_{s|F(D_1) \cap F(D_2)|}, w_{s|F(D_1) \cap F(D_2)|}^1, w_{s|F(D_1) \cap F(D_2)|}^2)\} \quad (4)$$

The similarity between the two tables, D_1 and D_2 , is computed by equation (5),

$$sim(D_1, D_2) = \frac{2(\sum_{i=1}^{|F(D_1) \cap F(D_2)|} w_{si}^1 + \sum_{i=1}^{|F(D_1) \cap F(D_2)|} w_{si}^2)}{\sum_{i=1}^{|D_1|} w_{1i} + \sum_{i=1}^{|D_2|} w_{2i}}. \quad (5)$$

The value which is computed by equation (5), is always given as a normalized value between zero and one, as shown in equation (6),

$$0 \leq sim(D_1, D_2) \leq 1. \quad (6)$$

Let us make some remarks on the encoding process and the computation of the similarity between texts. A text is encoded into a table by the text indexing, the word weighting, and the entry selection. A table is expressed as a set of entries, and a table is filtered into a set of words by the defined function. The similarity between tables is computed based on shared words by equation (5). In the future research, we consider representing a text into multiple tables.

B. Initial Version of Table based KNN Algorithm

This section is concerned with the initial version of table based KNN algorithm. It was initially proposed as the approach to the text classification by Jo in 2015 [5]. The similarity metric between tables which was described in the previous section is used for computing the similarity between a novice table and a training example. The labels of its nearest neighbors are voted with their identical weights for deciding the label of a novice input. This section is intended to describe the initial version of KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into tables, and they are notated by a set $Tr = \{T_1, T_2, \dots, T_N\}$. A novice word is encoded into a table notated by T . Its similarities with the tables which represent the sample words are computed by equation (5),

as $sim(T, T_1), sim(T, T_2), \dots, sim(T, T_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

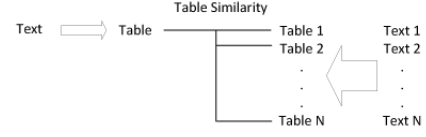


Figure 3. Similarities of Novice Input with Training Examples

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (7),

$$Nr = Select_k(Tr, T), Nr \subseteq Tr \quad (7)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (8),

$$Nr = \{T_1^{near}, T_2^{near}, \dots, T_k^{near}\} \quad (8)$$

The elements in the set, Nr , are used for voting their labels in the next step.

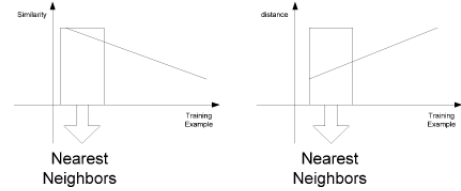


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = label(T_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $Count(Nr, c_j)$. The label of the novice item, T , is decided by equation (9),

$$label(T) = \arg\max_{j=1}^m Count(Nr, c_j) \quad (9)$$

The process of deciding the label of a novice item by equation (9), is called voting.

Let us make some remarks on the initial version of KNN algorithm from which we derive some variants. It computes the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice item is decided by voting the labels of

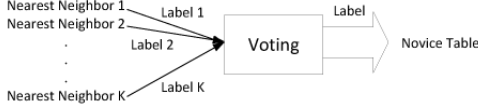


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. KNN Variants

This section is concerned with the variants which are derived from the KNN algorithm which was covered in Section III-B. The difference from the traditional version is to adopt similarity metric between tables for computing the similarity between a novice item and a training example. In this section, we derive the three variants by discriminating nearest neighbors, entries, or training examples. The three variants of KNN algorithm are adopted for implementing the text segmentation system. This section is intended to describe the three variants.

Let us mention the KNN variant which is derived by discriminating the nearest neighbors. A set of nearest neighbors is notated by $Nr = \{T_1^{near}, T_2^{near}, \dots, T_k^{near}\}$, and its elements are assumed as $sim(T, T_1^{near}) \geq sim(T, T_2^{near}) \geq \dots \geq sim(T, T_k^{near})$. The weights, $w_1, w_2, \dots, w_k$, are assigned to the nearest neighbors, $T_1^{near}, T_2^{near}, \dots, T_k^{near}$, following, $w_1 \geq w_2 \geq \dots \geq w_k$, and the total weights are computed for the category, c_j by equation (10),

$$CS(Nr, c_j) = \sum_{T_i \in Nr} w_i \quad (10)$$

The label of the novice item, T , is decided by equation (11),

$$label(T) = \arg\max_{j=1}^m CS(Nr, c_j) \quad (11)$$

There are two ways of assigning weights to the nearest neighbors: exponentially decreasing way and arithmetic decreasing way as shown in Figure 6.

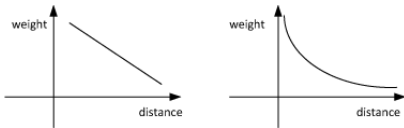


Figure 6. Discrimination over Nearest Neighbors

Let us mention the second table based KNN variant where the entries are discriminated for computing the similarity between a novice item and a training example as shown in Figure 7. The occurrences of the text, d_i , in the words which are labeled with the category, c_j , is $f(d_i|c_j)$, and the total occurrences of the text, d_i in the sample words is $f(d_i)$. The

influence of the text, d_i , on the category, c_j , is computed by equation (12),

$$I(d_i, c_j) = \frac{f(d_i|c_j)}{f(d_i)} \quad (12)$$

and the weight, w_i , is computed by equation (13),

$$w_i = \max_{j=1}^m I(d_i, c_j). \quad (13)$$

Equation (5) for computing the similarity between tables is modified into equation (14),

$$sim(T_1, T_2) = \frac{2 \left(\sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_i w_{s_i}^1 + \sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_i w_{s_i}^2 \right)}{\sum_{i=1}^{|T_1|} w_i w_{1i} + \sum_{i=1}^{|T_2|} w_i w_{2i}}. \quad (14)$$

In this variant, equation (5) is replaced by equation (14) for computing the similarity between a novice item and a training example.

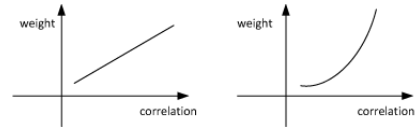


Figure 7. Discrimination over Table Entries

Let us mention the third KNN variant where the training examples are discriminated for computing the similarity between a novice item and a training example and/or voting the nearest neighbors for deciding the label as shown in Figure 8. The similarity threshold is used as the parameter, and for each training example, other training examples are taken within the threshold from it as its nearest neighbors. The number of nearest neighbors and the number of training examples which are labeled identically to the training example, T_i , are notated respectively by r_i and r_i^- , and the weight is computed by equation (15),

$$w_i = \frac{r_i^-}{r_i} \quad (15)$$

The similarity between the novice item, T , and the training example, T_i , is computed by equation (16),

$$sim(T_1, T_2) = w_i \left(\frac{2 \left(\sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_{s_i}^1 + \sum_{i=1}^{|F(T_1) \cap F(T_2)|} w_{s_i}^2 \right)}{\sum_{i=1}^{|T_1|} w_{1i} + \sum_{i=1}^{|T_2|} w_{2i}} \right), \quad (16)$$

and the total weight is computed for the category, c_j , by equation (17), in voting,

$$CS(Nr, c_j) = \sum_{T_i \in Nr} w_i. \quad (17)$$

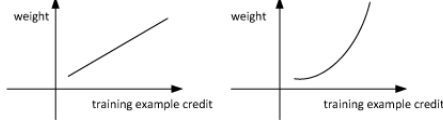


Figure 8. Discrimination over Training Examples

Equation (16) and (17) are used respectively for computing the similarity between a novice item and a training example and voting the nearest neighbors for each category.

Let us make some remarks on the three variants of KNN algorithm which are proposed as the approaches to the text segmentation. In the first variant, the nearest neighbors are discriminated for voting their labels. In the second variant, the entries are discriminated for computing the similarity between a novice input and a training example. In the third variant, the training examples are discriminated for voting the labels of the nearest neighbors and/or computing the similarity between tables. We may consider the trainable KNN algorithm which optimizes the weights of the nearest neighbors, the entries, and the training examples for minimizing the training error.

D. System Architecture

This section is concerned with the architecture and the execution flow of the text segmentation system. In Section III-C, we describe the table based KNN variants and adopt them for implementing the text segmentation system. The initial KNN algorithm which is mentioned in Section III-B is replaced by its variants in the architecture of the text segmentation system which was previously proposed. The process of executing the system is to encode an adjacent paragraph pair into a table and classify it into boundary or continuance. This section is intended to describe the text segmentation system which is implemented in this study.

The process of gathering sample paragraph pairs and classifying a novice one, is illustrated in Figure 9. Because even a same paragraph pair may be classified differently depending on the domain, the task is called domain dependent classification. Sample paragraph pairs are gathered domain by domain, and each pair is labeled with boundary or continuance. The paragraph pairs are taken from texts which are tagged with their same domain, each paragraph pair is classified into boundary or continuance. Sample paragraph pairs which are labeled with one of the two categories are encoded into tables in each domain.

The entire architecture of the proposed text segmentation system is illustrated in Figure 10. A text is given is the input, and paragraph pairs are generated from the text by sliding the two sized window. The sample paragraph pairs in the boundary group and the continuance group and novice paragraph pairs which are generated from the input text are encoded into tables, in the encoding module. The novice paragraph pairs are classified into one of the two categories

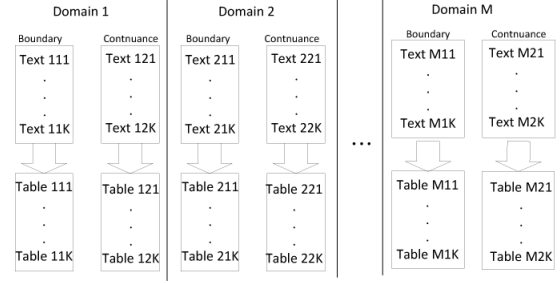


Figure 9. Preparation of Training Examples

in the similarity computation module and the voting module. The difference from the system architecture which was proposed in [7] is to replace the initial KNN algorithm by its variants.

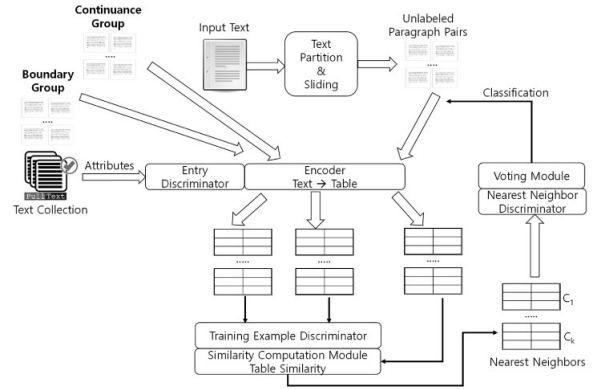


Figure 10. System Architecture

The execution process of the proposed system is illustrated as a block diagram in Figure 11. The sample paragraph pairs which are labeled with boundary or continuance are collected and encoded into tables. The paragraph pairs are generated from the input text by sliding the two-sized window on it, and they are encoded into tables. One among the three KNN variants is selected and applied to the classification of each paragraph pair into boundary or continuance. The boundary is marked between paragraphs in the pair which is classified into boundary.

Let us make some remarks on the proposed system whose architecture is illustrated in Figure 10. The M domains are predefined, in each domain, sample paragraph pairs each of which is labeled with boundary or continuance are prepared, and they are encoded into tables. The paragraph pairs which are generated from a text by sliding the two sized window are classified by one of the three KNN variants which are described in Section III-C. The difference from the previous version of text segmentation system is the ability to select one of the nearest neighbor discriminations, the entry discriminations, and the training example discriminations. The better performance of text segmentation is expected by

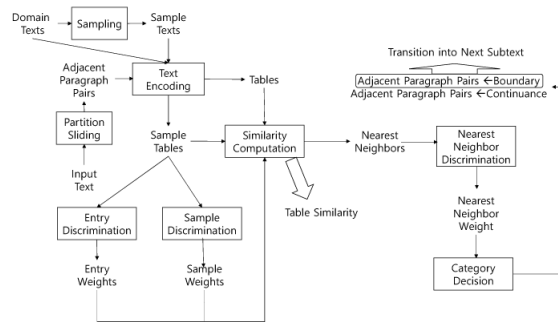


Figure 11. System Flow

replacing the initial version by its variants.

IV. CONCLUSION

Let us mention the significances of this research as the conclusion. We apply the proposed similarity metric to the KNN variants as well as the standard version. We derive the three variants from the standard version by discriminating the nearest neighbors, the entries, and the training examples. We design the text segmentation system by adopt the KNN variants with the proposed similarity metric. In the next research, we will implement it as a real system.

Let us mention the remaining tasks for improving this research. A text may be encoded into a compound representation which consists of more than one structured form, and a similarity metric between compound representations may be defined. We may modify other machine learning algorithms, such as the Naïve Bayes and the SVM (Support Vector Machine), into the table-based versions, using the proposed similarity metric between tables. The proposed KNN variants may be applied to the cooperation of this task with the text classification or the text clustering. The proposed KNN variants may be adopted for implementing the text segmentation system as an independent program or a library module.

REFERENCES

-
- Figure 11. System Flow
- replacing the initial version by its variants.
- #### IV. CONCLUSION
- Let us mention the significances of this research as the conclusion. We apply the proposed similarity metric to the KNN variants as well as the standard version. We derive the three variants from the standard version by discriminating the nearest neighbors, the entries, and the training examples. We design the text segmentation system by adopt the KNN variants with the proposed similarity metric. In the next research, we will implement it as a real system.
- Let us mention the remaining tasks for improving this research. A text may be encoded into a compound representation which consists of more than one structured form, and a similarity metric between compound representations may be defined. We may modify other machine learning algorithms, such as the Naïve Bayes and the SVM (Support Vector Machine), into the table-based versions, using the proposed similarity metric between tables. The proposed KNN variants may be applied to the cooperation of this task with the text classification or the text clustering. The proposed KNN variants may be adopted for implementing the text segmentation system as an independent program or a library module.
- #### REFERENCES
- [1] E.H. Han, G. Karypis and V. Kumar, “Text Categorization Using Weight Adjusted k-Nearest Neighbour Classification” pp53-64, in *Advances in Knowledge Discovery and Data Mining*. PAKDD 2001, Berlin, Heidelberg:Springer, 2035, 2001.
 - [2] T. Jo, “Table based Matching Algorithm for Soft Categorization of News Articles in Reuter 21578”, 875-882, *Journal of Korea Multimedia Society*, 11, 2008.
 - [3] H. Parvin, A. Hosein, and M.B. Behrouz, “MKNN: Modified k-nearest neighbor” *Proceedings of the World Congress on Engineering and Computer Science*. Vol. 1. Newswood Limited, 2008.
 - [4] T. Jo, “Device and Method for Categorizing Electronic Document Automatically”, 10-2009-0041272, 10-1071495, 2011.
 - [5] T. Jo, “Normalized Table Matching Algorithm as Approach to Text Categorization”, 839-849, *Soft Computing*, 19, 2015.
 - [6] T. Jo, “Content based Segmentation of Texts using Table based KNN”, 27-32, *The Proceedings of 16th International Conference on Advances in Information and Knowledge Engineering*, 2017.
 - [7] T. Jo, “Using Table based Version of K Nearest Neighbor for Segmenting News Articles by their Contents”, 62-64, *The Proceedings of 1st International Conference on Advanced Engineering and ICT-Convergence*, 2018.
 - [8] A.A. Nababan and O. S. Sitompul, “Attribute weighting based K-nearest neighbor using Gain Ratio”, *Journal of Physics: Conference Series*, 1007, 1, 2018.
 - [9] T. Jo, “Segmenting Long Texts Automatically by Table based K Nearest Neighbor”, in Preparation, 2023.